Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 18 (2021) 129-137

www.revista-riai.org

Principios básicos para el desarrollo de una aplicación de bi-manipulación de cajas por un robot humanoide

Hernandez-Vicen, J.*, Martinez, S., Balaguer, C.

Universidad Carlos III de Madrid, Avd. Universidad, 30, Leganés, Madrid, España.

To cite this article: Hernandez-Vicen, J., Martinez, S., Balaguer, C. 2021. Basic principles for the development of an application to bi-manipulate boxes with a humanoid robot. Revista Iberoamericana de Automática e Informática Industrial 18, 129-137. https://doi.org/10.4995/riai.2020.7133

Resumen

La logística es un sector que está en continuo crecimiento, debido tanto a la globalización, como a la actual situación creada por el Covid. En este artículo se describe una aplicación para reconocer cajas, extrayendo sus características con el fin de identificar la cara de apertura por medio de un sistema de visión por computador. Este objetivo se ha conseguido teniendo en cuenta las dimensiones y la posición en el espacio de la misma, logrando estas características a través de técnicas de procesamiento de imagen en 2D y en 3D. Posteriormente, la información correspondiente a las caras de la caja es clasificada con un árbol de decisiones, obteniendo así la probabilidad de que cada una de las seis caras sea la de apertura. Este artículo sirve para establecer las bases para desarrollar en un futuro una aplicación en la que el robot humanoide TEO mediante aprendizaje encuentre la forma más óptima de bimanipular cajas y abrirlas, integrando este conocimiento en un sistema automatizado.

Palabras Clave: Visión por computador, Corrección errores, Clasificación, Robot humanoide.

Basic principles for the development of an application to bi-manipulate boxes with a humanoid robot.

Abstract

Logistics is a sector which is continuously growing, due to globalization, as well as the current situation caused by the Covid. In this article, an application to recognize boxes is described. The characteristics are extracted with the goal of identify the opening side of the box by using computer vision techniques. This goal has been achieved considering the dimensions, as well as, the position in the space of the box. Those characteristics were obtained processing 2D and 3D images. Then, this information has been classified by using a decision tree based on the human knowledge. The probability of each of the six faces to be the opening side is obtained. This article is a base to develop in the future an application in which the humanoid robot TEO is capable to learn the optimal way to find the opening of boxes and bimanipulate them to be opened in an automated system.

Keywords: Computer vision, Errors correction, Classification, Humanoid robot.

1. Introducción

El comercio electrónico está aumentando a nivel mundial año tras año, superando en 2018 los 40.000M€ facturados en España (Galeano, 2019). El aumento de estas transacciones electrónicas está permitiendo el acceso de PYMES a mercados más lejanos y un incremento en las operaciones logísticas para poder suministrar los productos adquiridos por los clientes. Como consecuencia, en 2017 la logística suponía el 7,8% del PIB en España (García, 2017).

Todas las operaciones logísticas que se dan en un país son medidas y evaluadas periódicamente basándose en unas variables definidas por el Banco Mundial. Estas permiten medir el desempeño logístico de cada país, realizando un análisis comparativo a nivel mundial que queda reflejado en

*Autor para correspondencia: juanhernandezvicen@gmail.com

un ranking que recibe el nombre de: "Logistics Performance Index" (Índice de Desempeño Logístico) LPI. Entre el 2010 y el 2018 España ha escalado del puesto 25 a la posición 17 (World Bank Group, 2019), quedando reflejada la evolución que ha experimentado este sector. Sin embargo, para que continúe este ritmo de crecimiento, es necesario especializarse e investigar para desarrollar tecnologías que permitan optimizar procesos y acortar tiempos.

Actualmente, existen ya robots que realizan tareas de logística más enfocadas a carga, descarga, paletizado, despaletizado… (Echelmeyer, 2008). Por ejemplo, hay almacenes de Amazon que cuentan con plataformas móviles autónomas que transportan estanterías con productos a lo largo de la instalación. Para el reparto o envío del producto al cliente también existen soluciones como la entrega por medio de drones que plantea la misma empresa.

Aun así, siguen existiendo bastantes campos de mejora en la automatización de tareas en la logística: En actividades como la devolución de paquetes, actualmente es necesario recurrir a una persona que manipule la caja, encuentre la apertura y sea capaz de abrirla y clasificar lo que hay en su interior. Aprovechando esta carencia, se plantea una aplicación en la cual, combinando técnicas de visión artificial con técnicas de clasificación, un robot humanoide sea capaz de identificar una caja, y bimanipularla para encontrar de forma óptima la cara de apertura (Figura 1).

Para ello, se utilizará el robot humanoide TEO (Martínez, 2012), (Monje, 2011) del RoboticsLab de la Universidad Carlos III de Madrid, con el que ya se han realizado otras investigaciones en visión artificial (Hernandez-Vicen, 2018).

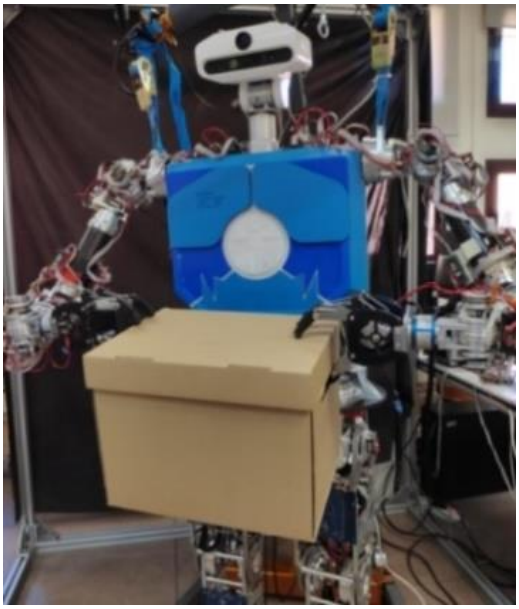

Figura 1: TEO bi-manipulando caja

En este artículo, se desarrolla un sistema de visión por computador que permite identificar la caja y sus características. Así como, distinguir y clasificar la cara de apertura con el fin de automatizar este proceso. Los datos que definen cuál es la cara de apertura, son clasificados mediante un árbol de decisiones, basado en el conocimiento humano. Se hace especial hincapié en el procesamiento de imágenes, ya que, el objetivo que se persigue es la obtención de datos de buena calidad, que faciliten la detección de la cara de apertura y su clasificación. Esta información servirá de base cuando el aprendizaje del robot se desarrolle. Así mismo, la precisión en la obtención de la posición de la caja y de su volumen, favorecerá el cálculo de la cinemática en las tareas de bimanipulación.

2. Condiciones de contorno y arquitectura de la aplicación

Aunque el objetivo global que se persigue es el desarrollo de una aplicación de bimanipulación, este artículo se centra en la extracción de información para distinguir la caja y en la clasificación de las caras para encontrar la apertura. Mediante visión artificial, se obtienen todos los datos necesarios para identificar las características que definen una caja de cartón. Esto incluye tanto características volumétricas o de forma, como posición de texto o logos en sus caras.

2.1 Condiciones de contorno

La obtención de estas características se ha realizado configurando un espacio en el que se disponen cajas de forma individual en el centro de la imagen con el fin de poder diferenciar su contorno exterior. Además, el fondo es uniforme, y se aplica un filtro de profundidad para eliminar el ruido en la escena. La iluminación es la propia del laboratorio, sujeta a los cambios de luz que entra por las ventanas. A pesar de que en un ambiente industrial se puede trabajar con un escenario de iluminación bastante más controlado, se ha intentado no llegar a ese extremo con el fin de que la aplicación final pueda tener una mayor versatilidad de aplicación en distintos entornos.

Tal y como se observa en la Figura 2, el origen de las coordenadas de referencia se ha situado en la cabeza de TEO (lugar donde se encuentra el sensor de visión).

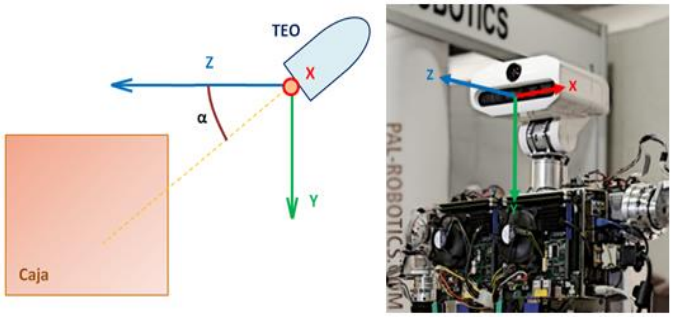

Figura 2. Origen de coordenadas en el espacio. Configuración de las condiciones de contorno.

2.2 Arquitectura de la aplicación

La arquitectura que se presenta en esta sección, describe los pasos seguidos para obtener la cara de apertura de la caja. Estos quedan reflejados de forma esquemática en la Figura 3.

Primero se han obtenido las imágenes en 2D y las nubes de puntos 3D utilizando la cámara Asus Xtion Pro que lleva instalada el robot humanoide TEO en su cabeza. A continuación, se ha postprocesado la información. En la imagen 2D se ha segmentado la caja y se han extraído los bordes y vértices de esta, distinguiéndola del resto de la imagen y obteniendo el número de caras visibles.

Además, de estas caras se han reconocido las palabras y logos, girando la imagen para saber la orientación de estos datos.

Respecto a la nube de puntos en 3D, se ha logrado la relación en mm de cada píxel en función de la profundidad a la que se encuentran los datos. Con los bordes de la caja extraídos del procesamiento de la imagen, la equivalencia de píxel/mm según la profundidad y aplicando una corrección de perspectiva teniendo en cuenta la orientación de la cámara de la cabeza de TEO, se ha calculado la posición de la caja en el espacio, sus dimensiones volumétricas e identificado sus caras.

Por último, con el fin de dar los primeros pasos para implementar en un futuro un sistema de inteligencia artificial, se ha introducido un algoritmo de clasificación basado en un árbol de decisiones que ha sido entrenado con la experiencia humana. Gracias a esto, se ha realizado una primera aproximación con la que el robot puede conocer cuál es la cara correcta de apertura de la caja. En un futuro, con esta información como base, podrá tomar la decisión de hacia qué lado debe girar la caja para encontrar la apertura de forma óptima.

Figura 3. Estructura de la aplicación.

3. Detección caja 2D

Para segmentar un objeto concreto diferenciándolo del resto de la imagen (Vázquez, 2015), es necesario conocer las características relevantes que lo definen. En el caso de una caja son: color, forma cúbica y aristas rectas. Primero se ha utilizado el color para separar la caja del resto de la imagen y a continuación, utilizando las características de las líneas rectas y su forma geométrica, se han obtenido los vértices de la caja.

Tal y como se ha expuesto en el apartado de condiciones de contorno, al introducir imágenes obtenidas directamente por la cámara de TEO, aparecen una serie de distorsiones en la imagen, siendo necesario simplificar el fondo para reducir este ruido. Para evitarlo, se ha realizado el experimento con una tela negra uniforme detrás de la caja. A pesar de esto, sigue habiendo ruido causado por la iluminación. Por ello, se procede a separar la imagen en los 3 canales del espacio RGB realizando una umbralización sobre el canal rojo.

En pseudocódigo el procesado de la imagen en 2D queda explicado de la siguiente forma (Algoritmo 01):

Algoritmo 01: Obtención vértices caja (Procesamiento 2D).

Entrada: Captura imagen
Salida: Vértices (V_0 a V_5)

Se requiere: Cámara encendida

```
1:  //Pre-procesado imagen
2:  Captura (Img)
3:  Separar (Img, canales RGB)
4:  Operación And (Img & Canal R) // Suma imagen original e imagen en canal R
5:
6:  //Obtención caras y vértices de la caja
7:  Canny (Img_preproc)
8:  HoughLinesP (Img_preproc) //Obtención líneas finitas
9:  Dilate (Img_preproc) //Aumentar grosor de las líneas para fusionarlas
10: Erode (Img_preproc) //Reducir grosor líneas al original
11: Squarecontours (Img_preproc) //Búsqueda formas cuadradas
12: Vértices(Img_preproc)
13: if (nº Vértices pos (X,Y) similar >=2)
14:     Media (Vértices pos similar (X,Y))
15: end if
16: Nº Vértice (Pos(X,Y), vértice)
17: Salida (V0 a V5)
```

De esta forma, solo se distingue en la imagen la zona correspondiente a la caja (Figura 4, izquierda). A continuación, realizando una operación AND se combinan los píxeles blancos resultantes de la umbralización junto con la imagen original. Como resultado, se logra una imagen combinada en la que se distingue la caja pero el fondo está en píxeles negros. De esta forma se obtiene la caja sin ningún ruido originado por el fondo.

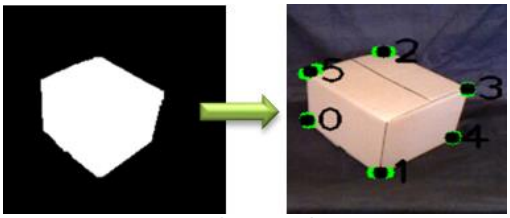

Figura 4. Segmentación caja imágenes reales 2D

Después, se procede a la extracción de las aristas de la caja, utilizando las librerías de OpenCV (Bradski, 2008a), (Bradski, 2000b). Sobre la imagen ya preprocesada se ha aplicado un filtro canny, dejando la imagen binaria y resaltando los bordes en blanco y en negro de la caja. Acto seguido, se ha utilizado la transformada de Hough (Brahmbhatt, 2013), (Elfring, 2013). Concretamente se ha utilizado el método HoughLinesP, mediante el cual se

obtienen rectas finitas con un origen y un final definido (OpenCV: Hough Line Transform., 2020).

Siguiendo estos pasos, tal y como se aprecia en la imagen, surge un problema: se consiguen varias rectas abiertas por cada arista (Figura 5.1). Para corregirlo, se aplican operaciones de erode/dilate cerrando la forma de la caja (Figura 5.2).

Teniendo el contorno de la caja cerrado, se ha procedido a identificar cuadrados por cada cara visible. Para ello, se ha utilizado un buscador de contornos que busca figuras geométricas cuadradas (Figura 5.3).

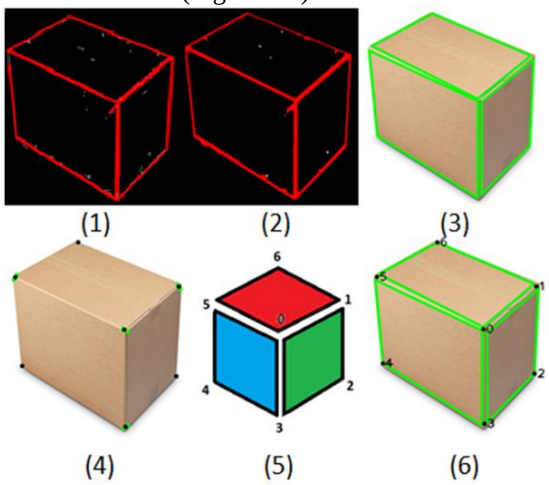

Figura 5. Proceso segmentación caja 2D

Tras la obtención de las caras, se han identificado los vértices de la caja para poder conocer los límites. Además, en las esquinas donde se detecta más de un punto por vértice (por la coincidencia de varias caras), se aplica una media, disponiendo el punto en esa posición, aproximándolo a la solución más real. (Figura 5.4).

Finalmente, con los vértices extraídos, se procede a ordenarlos para poder combinarlos con la información 3D con el objetivo de construir el volumen de la caja y posicionarla en el espacio (Figuras 5.5 y 5.6).

4. Relación píxel-mm según la profundidad.

Para saber el tamaño real de los objetos captados por la imagen, es necesario conocer la relación entre un píxel y el tamaño de ese objeto en función de la profundidad. Normalmente, se suele utilizar un patrón como referencia (De la Escalera, 2010), (Wang, 2010). En este caso, para obtener la relación entre la imagen y la realidad se ha utilizado como patrón un folio DIN A4, por su medida estandarizada (210x297mm), sobre un fondo oscuro uniforme dispuesto de forma totalmente perpendicular a la cámara. Aprovechando la diferencia de color y la uniformidad del fondo se segmenta este patrón del resto de la imagen y se capturan datos del número de píxeles horizontales y verticales en función de la profundidad. Se ha considerado un rango de profundidad de 600 a 750mm, distancia a la que se encontrará la caja para facilitar la manipulación en un futuro. Se han realizado barridos variando la profundidad 10mm dentro del rango establecido, obteniendo las siguientes gráficas que representan la relación pixel/ancho (Figura 6) y pixel/alto (Figura 7) en función de la profundidad:

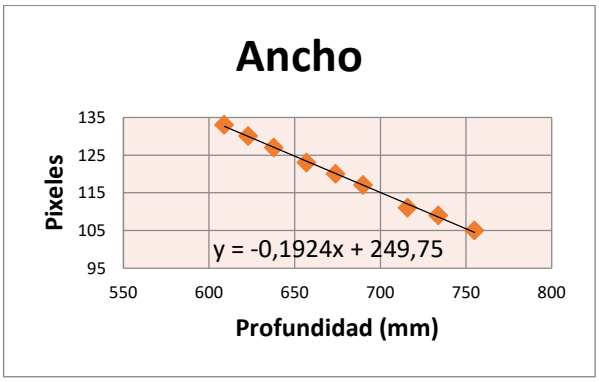

Figura 6. Relación entre el ancho real y píxeles en función de la profundidad.

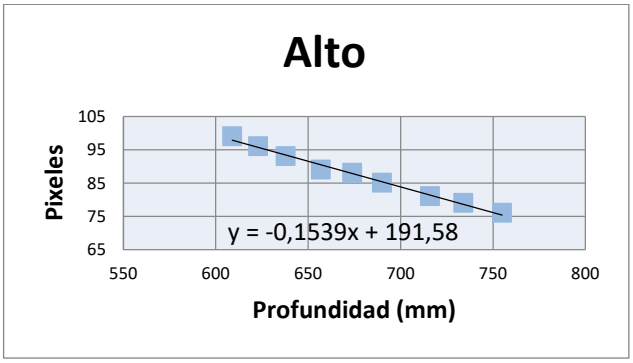

Figura 7. Relación entre altura real y píxeles según la profundidad.

$$y_{ancho} = -0{,}1924X + 249{,}75 \quad (1)$$

$$y_{alto} = -0{,}1539X + 191{,}58 \quad (2)$$

Utilizando las ecuaciones (1) y (2) resultantes de los datos representados en las gráficas pertenecientes a las Figuras 6 y 7, e introduciendo el valor de *X*, donde *X* es la profundidad de lectura de los píxeles detectados, se calcula el tamaño real de este píxel.

5. Procesamiento 3D y obtención de datos medibles.

Con la relación entre los milímetros que equivale cada pixel en función de la profundidad de la imagen, así como con la información obtenida de las imágenes en 2D (posición de los vértices en la imagen), se hace posible calcular la posición real de la caja en el espacio, así como sus dimensiones reales y volumen.

5.1. Obtención de la medida real del eje X

Para la obtención de esta medida, lo primero que se hace es encontrar un punto de interés en el espacio de la imagen y conseguir la profundidad a la que se encuentra. En el caso de la caja, sirve cualquiera de los vértices de ésta.

Tras obtener la profundidad de ese punto, se sustituye el valor de *X* en la fórmula obtenida a partir de la gráfica de la Figura 6. Así conocemos la equivalencia del ancho en mm de cada píxel para esa profundidad dada. Quedando la fórmula de la siguiente forma:

$$Píxeles_{ancho} = prof * (-0{,}1924) + 249{,}75 \quad (3)$$

Conocida esta relación y sabiendo la distancia desde el punto de interés (vértice de la caja) al centro de la imagen en

el eje horizontal en píxeles, podemos calcular la longitud real del centro de la imagen al punto de interés aplicando la ecuación 4:

$$Distancia_{real}\,(mm) = Píxeles_{ancho} * nºpíxeles_{cero\text{-}centro} \quad (4)$$

5.2. Obtención de la medida real del eje Y y del eje Z

A continuación, se calcularán la distancia real en los otros dos ejes. Para ello, esta vez se utilizará la fórmula obtenida en la Figura 7. Sin embargo, para obtener la distancia no basta con aplicar simplemente la fórmula como en el caso anterior, sino que es necesario implementar una corrección por trigonometría. Ya que hay que tener en cuenta que la cámara de TEO está instalada en su cabeza, a un ángulo que varía según el ángulo α que esté inclinada respecto a la horizontal.

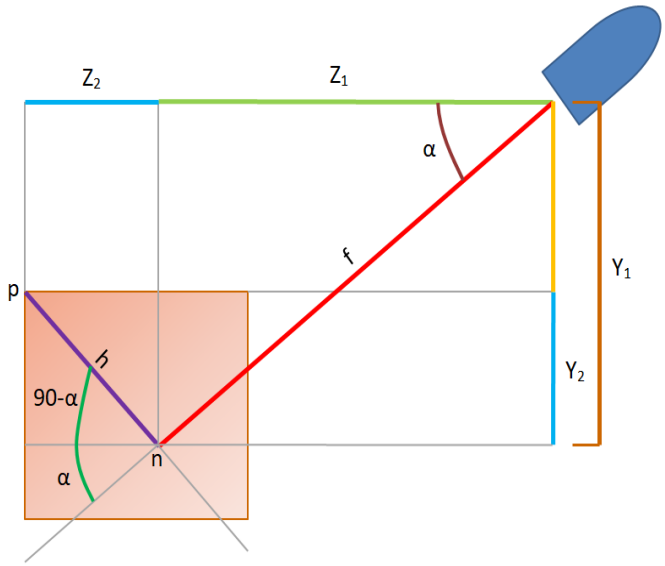

Figura 8. Descomposición trigonométrica ejes Y y Z.

En la Figura 9 viene representada la silueta de la cabeza del robot (esquina superior derecha) y la caja de cartón (esquina inferior izquierda). La recta diagonal que cruza la imagen representa la profundidad f a la que se encuentra el plano que contiene el punto p tomando como origen la cámara. En los siguientes pasos se muestran el proceso de cálculo para lograr la distancia real de la cámara descompuesta en ambos ejes desde la cámara de TEO (referencia de origen) hasta el vértice "p".

La primera operación es descomponer mediante ecuaciones trigonométricas el plano de profundidad f en el que se encuentra nuestro punto de interés p, teniendo en cuenta el grado de inclinación de la cámara α, con esto obtenemos Z_1 e Y_1:

$$Z_1 = f * cos \quad (5)$$
$$Y_1 = f * sin \quad (6)$$

Los datos obtenidos en Z_1 e Y_1 representan la posición real en ambos ejes del píxel central de la imagen obtenida por la cámara. Sin embargo, los puntos de interés (vértices) rara vez suelen coincidir con la posición central, habiendo una diferencia de posición llamada "*h*". Para calcular esta distancia real se utiliza la fórmula obtenida en la Figura 7 de calibración:

$$Píxeles_{alto} = f * (-0{,}1539) + 191{,}58 \quad (7)$$

Conociendo la relación píxeles/mm y la distancia en píxeles, obtenemos la distancia real "*h*":

$$h(mm) = Píxeles_{alto} * nºpíxeles_{p\text{-}centro} \quad (8)$$

Tras obtener la distancia *h*, y conociendo el ángulo de inclinación de la cabeza se obtiene Z_2 e Y_2:

$$Z_2 = h * cos(90-\alpha) \quad (9)$$
$$Y_2 = h * sin(90-\alpha) \quad (10)$$

Finalmente, sumando Z_1 y Z_2 se obtiene la distancia real en mm en el eje Z y sumando Y_1 e Y_2 se calcularía en el eje Y, teniendo datos reales del tamaño de la caja, su posición real en el espacio y volumen.

5.3. Error al obtener las dimensiones de la caja.

Tras las operaciones llevadas a cabo en los apartados anteriores, después de haber combinado la información obtenida en la imagen 2D con la nube de puntos en 3D, es posible obtener los datos correspondientes al largo, alto y profundidad de la caja.

Para conocer el error de cálculo que se genera al utilizar este desarrollo, se ha hecho un experimento:

Se han posicionado tres cajas de forma individual, cada una de ellas con un volumen diferente. A continuación, se han tomado 20 medidas de cada una y se ha obtenido el máximo, mínimo y la media del valor calculado en cada una de las tres dimensiones que forman el volumen (largo, alto y profundidad). Estos datos han sido expuestos en la Figura 9:

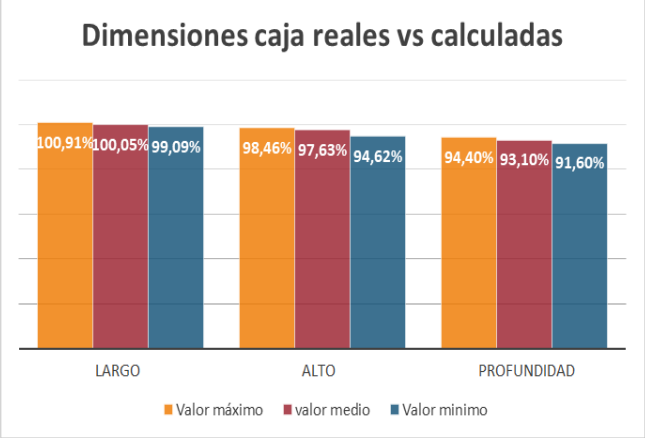

Figura 9. Error entre dimensiones reales y calculadas de la caja.

Tal y como se aprecia en la gráfica, el valor medio del largo tiene un error del 0.05%, el alto tiene un error del 2,37% y en la profundidad se observa un error del 6,9%. Estos resultados tienen lógica, ya que el largo y el alto, al obtenerse prácticamente de forma directa de la imagen 2D con la nube de puntos de profundidad, se introduce menos error que en la profundidad, dato que se obtiene realizando varios cálculos trigonométricos. Este error es aceptable, puesto que en el caso más desfavorable, un error del 6,9% en una caja estándar de profundidad de 250mm, supone un error de cálculo de 17,25mm, inferior a 2cm, no afectando a la manipulación.

6. Reconocimiento texto y logos.

En los apartados anteriores se han explicado los pasos para obtener características básicas de la caja: sus dimensiones, caras y posicionamiento en el espacio. Sin embargo, conocer

esta información es insuficiente para llegar a una conclusión de cuál es la cara de apertura adecuada de la caja.

Para encontrar la cara de apertura correcta de la caja, se ha comparado el caso robótico con el caso humano. Ante una caja lisa al no haber rasgos que permitan identificar la orientación de la caja, las personas deben escoger de forma aleatoria entre la cara superior e inferior donde se encuentran los cierres con cinta. El problema es que al abrir la caja no saben si el producto esta boca arriba o boca abajo. Al sistema robótico le ocurre lo mismo.

Sin embargo, al realizar el mismo experimento con una caja con logos/ imágenes/ texto en las caras laterales, el ser humano, en base a su conocimiento, abre la caja por la cara superior, en la cual pueden verse las imágenes rectas o se puede leer el texto. Cabe esperar que el fabricante introduzca su producto en la caja de tal forma que se pueda leer bien el texto o identificar el logo de su empresa. Dado que el texto y los logos o imágenes pueden ayudar a identificar la cara correcta de apertura para evitar dañar el producto, es necesario encontrar esta información, y saber distinguir su orientación para que el sistema robótico pueda localizar la cara de apertura correcta con mayor certeza.

Para la detección del texto (Figura 10), se han comparado tres estrategias diferentes. Escogiendo la más óptima para la solución del problema:

- OpenCV con librerías de detección de texto en imágenes Tesseract (Tesseract-ocr/tesseract, 2020) con resolución inicial (320x240píxeles): La principal ventaja que presenta es que, al no utilizar una resolución elevada, el coste computacional es menor, debido al menor tamaño de la matriz de la imagen. Sin embargo, presenta mayor dificultad a la hora de distinguir las palabras, requiriendo un aumento en el número de iteraciones que, como consecuencia, ralentizan el programa.

Figura 10. Detección texto en caja.

- OpenCV con librerías de detección de texto en imágenes con alta resolución (640x480píxeles): Se cambia la resolución de la imagen a la máxima que puede ofrecer la cámara y el texto es reconocido en la primera iteración. Sin embargo, este aumento en la resolución, al incrementar el número de píxeles por imagen, implica una matriz cuatro veces mayor, haciendo que los cálculos aumenten y el coste computacional se dispare. Esto implica un aumento en el tiempo de respuesta del robot, lo cual hace inviable esta solución.
- API de Google de visión: Esta librería usa tecnología de modelos de aprendizaje profundo (deeplearning), aprovechando la potencia que ofrece google, dado el amplio acceso que tiene a imágenes/logos… Al utilizarla, se observa que el tiempo de detección disminuye considerablemente respecto a las otras dos opciones, proporcionando además información más fiable.

Tabla 1. Tiempo detección texto (s)

DETECCIÓN TEXTO	*Res. Baja*	*Res. Alta*	*API google*
Tiempo por iteración	1t	4t	0.5t
Nº Iteraciones	15-20	1	1

Tras haber estudiado las tres opciones (Tabla 1), se acaba utilizando la librería de la API de google para obtener las características. Sin embargo, esto no es suficiente, ya que, la detección de textos o logos en sí no tiene interés. La información que realmente necesita tener el sistema es la orientación de estos datos en cada cara, para a partir de ahí sacar las conclusiones pertinentes de dónde se encuentra la cara adecuada de apertura. Para tener este dato, el programa escrito en pseudocódigo queda acorde al algoritmo 02:

Algoritmo 02: Orientación texto y logos en caras de la caja

Entrada: Imagen postprocesada. Caras visibles de la caja.
Salida: Orientación caras (0º, 90º, 180º, 270º)

```
1:  // Información obtenida del algoritmo 01
2:  Obtención caras (Img)
3:
4:  //Detección texto y logos en caras
5:  Detección texto/logos (Img) //Api Google
6:  if (texto o logos == true)
7:      Print (Orientación caja = 0º)
8:  else if (texto o logos == false)
9:      Rotar 90º (Img)
10:     Detección texto/logos (Img)
11:     if (texto o logos == true)
12:         Print (Orientación caja = 90º)
14:     else if (texto o logos == false)
15:         Rotar 90º (Img)
16:         Detección texto/logos (Img)
17:         if (texto o logos == true)
18:             Print (Orientación caja = 180º)
19:         else if (texto o logos == false)
20:             Rotar 90º (Img)
21:             Detección texto/logos (Img)
22:             if (texto o logos == true)
23:                 Print (Orientación caja = 270º)
24:             else if (texto o logos == false)
25:                 Print (No se han detectado texto o logos en
                    la caja.)
26:             end if
27:         end if
28:     end if
29: end if
17: Salida (Orientación)
```

Tal y como se observa en el algoritmo 02, se procesan las caras de la caja obtenidas en el algoritmo 01. Para ello, se utiliza la API de Google, aplicando la detección de texto y logos. Esta librería, tiene la limitación de que solo detecta el texto que se encuentra en horizontal en la imagen. Por este motivo, para detectar si hay información en esa cara es necesario hacer 4 iteraciones girando la imagen. En caso de que en la imagen inicial no se detecte información, se gira 90 grados y se vuelve a procesar con la librería de detección de texto y logos. Este giro se realiza dos veces más en el mismo sentido en caso de que no se detecten datos. Si se detecta información en uno de esos giros, como se conoce el número de iteraciones que se han realizado, se obtiene la orientación a la que se encuentra esa cara. Esto servirá de gran utilidad para estimar la cara de apertura.

7. Árbol de decisiones

Una vez extraídas las características de la caja, se ha realizado una primera aproximación para encontrar la cara de apertura de la caja utilizando estos datos. En este caso, se ha aplicado un algoritmo de clasificación basado en árboles de decisiones (Millán, 2006), (Sempere, 2014). Se ha escogido este método porque se dispone de información que puede ser clasificada de forma categórica: Existencia de texto, logos y de línea de apertura. Estos árboles de clasificación están basados en la teoría de árboles de clasificación y regresión desarrollada por (Breiman, 2017).

Se han seleccionado los 3 rasgos principales que se pueden diferenciar en la cara de una caja: Línea de apertura, letras y logo. En la Figura 11, se puede ver el árbol de decisiones creado para el texto y los logos, en el cual se clasifican los datos en función de su orientación. El árbol creado para la línea de apertura (Figura 12) es más simple, ya que solo existen dos opciones (que la cara cuente con esta línea o no).

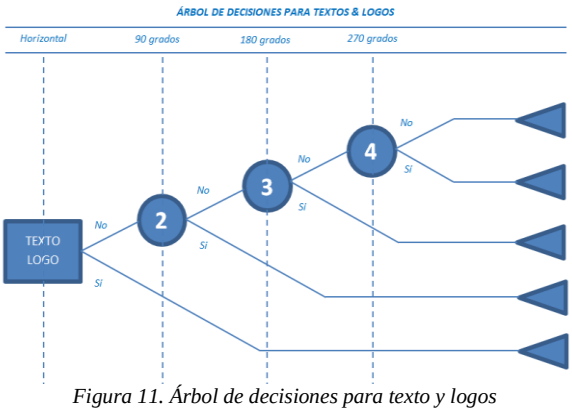

Figura 11. Árbol de decisiones para texto y logos

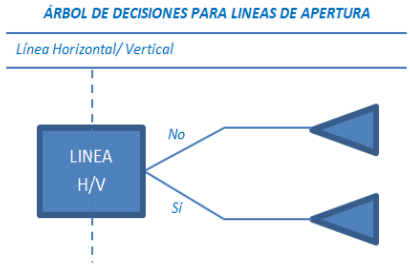

Figura 12. Árbol de decisiones para líneas de apertura

Tal y como se indica al comienzo del apartado de reconocimiento de texto y logos, esta información permite al ser humano identificar la cara de apertura correcta de la caja con mayor probabilidad de acierto. Siguiendo esta lógica, basada en la experiencia humana, a cada una de las opciones del árbol de decisiones se le atribuye un peso. La información relativa a las líneas de apertura tiene un peso de 0.6, mientras que la orientación del texto y de los logos, representa un 0.2 de peso por cada dato. Se ha dado un mayor peso a la detección de la línea puesto que es un factor mucho más determinante para encontrar la apertura que los otros dos datos.

Para obtener la cara de apertura, se ejecutan estos tres árboles de decisiones en cada una de las 6 caras de la caja, siguiendo el orden indicado en la Figura 13:

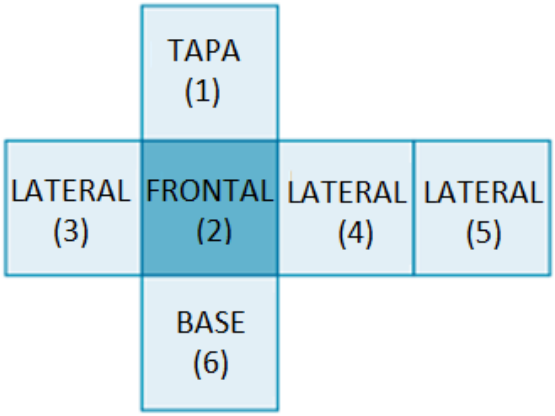

Figura 13. Caras de las cajas.

A medida que se procesan los datos de cada cara de la caja en los 3 árboles de decisión, se acumulan las probabilidades de qué cara de las 6 es la de apertura. Finalmente, tras haber obtenido toda la información, el algoritmo proporciona la cara con mayor probabilidad de ser la correcta para abrir la caja.

8. Experimentos y resultados

Para comprobar la eficacia del árbol de decisiones, se ha probado utilizando 3 cajas de diferentes tamaños y con distinta información en los laterales. Estas se han posicionado de forma individual en la zona frontal del robot, en el área de trabajo de éste (entre 600 y 750mm de distancia). En las Figura 14 se muestran tres cajas utilizadas para llevar a cabo los experimentos:

Experimento 01

Experimento 03.A

Experimento 02 Experimento 03.B

Figura 14. Cajas seleccionadas en los experimentos.

En la Figura 15, se muestran cada una de las seis caras que forman la caja seleccionada para el segundo experimento, estas caras siguen el orden establecido en la Figura 13.

CARA 1 *CARA 2* *CARA 3*

CARA 4 *CARA 5* *CARA 6*

Figura 15. Cara 1 a 6 de la segunda caja.

En el experimento se han dispuesto las cajas en distintas posiciones. Para comprobar si el algoritmo es capaz de llegar a la solución de la cara correcta de apertura, la segunda y la tercera caja se han posicionado de dos formas diferentes (EXP 02.A y 02.B) y (EXP 03.A y 03.B), orientándolas de distinta manera. En la Tabla 2, se presentan los resultados obtenidos con la probabilidad de acierto en cada cara (C1-C6) para cada experimento (EXP):

Tabla 2. Resultados de las pruebas.

EXP	C.1	C.2	C.3	C.4	C.5	C.6	AC.
01	0.24	0.21	0.29	0.29	0.18	0.16	NO
02.A	0.34	0.32	0.16	0.14	0.25	0.24	SI
02.B	0.17	0.15	0.46	0.36	0.16	0.17	SI
03.A	0.46	0.37	0.35	0.35	0.32	0.24	SI
03.B	0.25	0.46	0.29	0.29	0.36	0.28	SI

Tal y como se puede observar, tras procesar toda la información, el algoritmo ha acertado la cara correcta de apertura en cuatro de los cinco experimentos. Fallando el algoritmo en el ejemplo de la primera caja. El motivo de este fallo es que el sistema de visión ha detectado una raya central en cuatro de las seis caras, haciendo insuficiente la información transferida por parte de la observación humana.

Además, al tratarse de un de un algoritmo de clasificación, no acumula la información obtenida previamente, fruto de las iteraciones que va realizando. Por este motivo, la probabilidad de acierto que proporciona es tan baja. Como consecuencia de esto, aunque acierta en los otros cuatro casos, la tasa de acierto del algoritmo no es alta.

La tasa de aciertos, así como, el aumento de la probabilidad a la hora de escoger la cara de apertura, será mejorada en futuros trabajos cuando se incorpore un algoritmo más complejo basado por ejemplo en aprendizaje por refuerzo (Rebollo, 2009).

Por último, cabe destacar que está siendo tan acertado en la predicción porque está viendo las 6 caras de la caja. El objetivo del trabajo presentado en este artículo es que esta información sirva de punto de partida para desarrollar un sistema más complejo que permita identificar la cara de apertura sin tener que ver las 6 caras.

9. Conclusiones

Como conclusión, en el desarrollo de este trabajo, se han post-procesado imágenes en vivo por técnicas de visión artificial. En este caso, la caja se ha segmentado del entorno donde se encontraba y se ha extraído toda la información aprovechable y ha sido tratada. Estos datos han servido para distinguir y caracterizar las distintas caras de la caja.

Finalmente, se ha planteado una primera aproximación para detectar la cara de apertura de la caja, basándonos en la experiencia humana, y utilizando un árbol de decisiones. Si bien es cierto que la seguridad de éxito que obtiene el sistema es de entorno a un 50%, también se puede afirmar que la tasa de acierto de los experimentos que se han realizado está en torno al 80%.

De los resultados obtenidos, se deduce que esta primera aproximación tiene una tasa de fallo que no asegura el éxito para todos los tipos de casos que se pueden encontrar. Es decir, en cajas que se han considerado como estándar (con apertura por la mitad de una cara y el texto orientado hacia esta apertura), el funcionamiento es el adecuado. Sin embargo, cuando alguna de las caras presenta algún patrón extraño, complica el reconocimiento e impide que el algoritmo sea capaz de identificar bien la cara de apertura. Por ello, se debe mejorar el proceso y sustituir el algoritmo de clasificación por uno de aprendizaje más desarrollado que permitan a TEO ir aprendiendo de la experiencia adquirida.

Como trabajos futuros, tal y como se ha dicho anteriormente, los pasos dados hasta ahora abren la puerta a la necesidad de la utilización de un algoritmo de aprendizaje para poder mejorar la aplicación, que cumpla con dos puntos: por un lado, que sea capaz de ir acumulando la experiencia adquirida y que con esto vaya modificando sus futuras decisiones. Por otro lado, que sea capaz de identificar la apertura correcta de la caja sin tener que ver todas las caras y post-procesar todas las imágenes, ya que, actualmente para llegar a la conclusión de qué cara es la de apertura, necesita haber visto las 6 caras de la caja. Para aplicar este algoritmo más avanzado, habrá que estudiar los algoritmos existentes, seleccionar el que se adecue más según la tipología de datos que tenemos, adaptarlo a las necesidades de la aplicación, desarrollar la lógica de funcionamiento e implementar este

algoritmo con los datos obtenidos en esta investigación. Además, la información obtenida relativa a la posición y volumen de la caja, servirá también de punto de partida para obtener las posiciones agarre de la caja para su bimanipulación.

Agradecimientos

Este trabajo ha sido realizado parcialmente gracias al apoyo de RoboCity2030-DIH-CM; Madrid Robotics Digital Innovation Hub, S2018/NMT-4331, fundado por “Programas de Actividades I+D en la Comunidad de Madrid” y cofundado por los fondos estructurales de la UE.

Referencias

Galeano, S. 2019. El ecommerce español se quedó a las puertas de la barrera psicológica de los 40.000 millones de euros en 2018. 2019, de marketing4ecommerce Sitio web: https://marketing4ecommerce.net/facturacion-anual-ecommerce-espanol-no-supera-40-000-mme-cnmc-2018/

World Bank Group, 2019. Country Score Card: Spain 2018. Recuperado el 12/2019, de World Bank Group Sitio web: https://lpi.worldbank.org/international/scorecard/line/254/C/ESP/2018/C/ESP/2016/C/ESP/2014/C/ESP/2012/C/ESP/2010/C/ESP/2007#chartarea

García Juez, I. 2017. La logística y el transporte en España suponen el 8% del PIB y y emplean a 800.000 personas. Ok Diario. Recuperado el 12/2019 de https://okdiario.com/economia/sector-logistica-transporte-espana-supone-8-del-pib-emplea-800-000-personas-1368706

Echelmeyer, W., Kirchheim, A., & Wellbrock, E. (2008, September). Robotics-logistics: Challenges for automation of logistic processes. In 2008 IEEE International Conference on Automation and Logistics (pp. 2099-2103). IEEE.

Martínez, S., Monje, C. A., Jardón, A., Pierro, P., Balaguer, C., & Munoz, D. (2012). Teo: Full-size humanoid robot design powered by a fuel cell system. Cybernetics and Systems, 43(3), 163-180.

C. A. Monje, S. Martínez, A. Jardón, P. Pierro, C. Balaguer and D. Muñoz, "Full-size humanoid robot TEO: Design attending mechanical robustness and energy consumption," 2011 11th IEEE-RAS International Conference on Humanoid Robots, Bled, 2011, pp. 325-330. doi: 10.1109/Humanoids.2011.6100835

Hernandez-Vicen, J., Martinez, S., Garcia-Haro, J., & Balaguer, C. (2018). Correction of visual perception based on neuro-fuzzy learning for the humanoid robot TEO. Sensors, 18(4), 972.

Vázquez, E. (2015). Técnicas de visión artificial robustas en entornos no controlados. Tesis doctoral, Universidad de Vigo, Vigo, España. Recuperado el 13 de Enero de 2020, de https://dialnet.unirioja.es/servlet/dctes?codigo=124604

Bradski, G., & Kaehler, A. (2008a). Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc.".

Bradski, G., & Kaehler, A. (2000b). OpenCV. Dr. Dobb’s journal of software tools, 3.

Brahmbhatt, S. (2013). Practical OpenCV. Apress.

Elfring, J. (2013). Image Processing Using OpenCV. Online (Feb 2018).

OpenCV: Hough Line Transform. (2020). Recuperado 20 Enero 2020, de https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html

de la Escalera, A., Armingol, J. M., Pech, J. L., & Gómez, J. J. (2010). Detección Automática de un Patrón para la Calibración de Cámaras. Revista Iberoamericana de Automática e Informática Industrial RIAI, 7(4), 83-94.

Wang, Y. M., Li, Y., & Zheng, J. B. (2010, June). A camera calibration technique based on OpenCV. In The 3rd International Conference on Information Sciences and Interaction Sciences (pp. 403-406). IEEE.

Tesseract-ocr/tesseract. (2020). Recuperado el 20 Enero 2020, de https://github.com/tesseract-ocr/tesseract.

Shah, P., Karamchandani, S., Nadkar, T., Gulechha, N., Koli, K., & Lad, K. (2009, November). OCR-based chassis-number recognition using artificial neural networks. In 2009 IEEE International Conference on Vehicular Electronics and Safety (ICVES) (pp. 31-34). IEEE.

Millán, D. B., Boticario, J. G., & Viñuela, P. I. (2006). Aprendizaje automático. Sanz y Torres.

Sempere, J. (2014). Aprendizaje de árboles de decisión. Universidad Politécnica de Valencia, Valencia.

Breiman, L. (2017). Classification and regression trees. Routledge.

Rebollo, F. F., & Barrojo, D. (2009). Aprendizaje por Refuerzo. Aprendizaje Automático, Departamento de Informática, Escuela Politécnica Superior.